Biodata

Nama : Ari Cahyono
No-Reg : 149368582101-638
Kelas : DTS Scalable Web Service With GOLANG

Panduan Pembuatan Program:

1. Buat Folder Mygram
2. buka terminal dan jalankan go mod init Mygram
3. buat file main.go
4. instal beberapa package
    - GORM : untuk ORM
    - GIN : untuk Routing
    - dto-mapper : untuk membantu convert model ke DTO
    - govalidator : untuk membantu dalam validati data
    - jwt-go : untuk membantu dalam membuat token JWT
5. Buat beberapa folder
    - controllers
    - models
    - routers
    - middlewares
    - configs
    - dto
    - helpers
    - temp
6. buat model
    - user
    - photo
    - comment
    - social media
7. buat file pada folder config dengan nama config db.go dan daftarkan model yang digunakan
8. buat beberapa responseDtO pada folder dto
9. buat beberapa helper pada folder helper
10. buat controller pada folder controller, kemudian gunakan dto dan beberapa helpers yang sudah di buat pada setiap func di controller.
    - userControllers.go (CRUD)
    - photoControllers.go (CRUD)
    - commentControllers.go (CRUD)
    - socialmediaControllers.go (CRUD)
11. buat file authentication.go pada folder middlewares
12. buat file pada folder router dengan nama router.go buat endpoint api, hubungkan setiap endpoint dengan controller yang digunakan dan batasi beberapa endpoint dengan middlewares yang di butuhkan
13. Selesai.

Panduan Menjalankan Program:

BE:

1. Siapkan Database Postgres,
   kemudian buat Database dengan nama
   - mygram
2. jalankan program dengan perintah
   - go run main.go

Referensi / Package yang digunakan:

- GORM : untuk ORM
- GIN : untuk Routing
- dto-mapper : untuk membantu convert model ke DTO
- - govalidator : untuk membantu dalam validati data
- - jwt-go : untuk membantu dalam membuat token JWT

Postman :

URL Collection API :

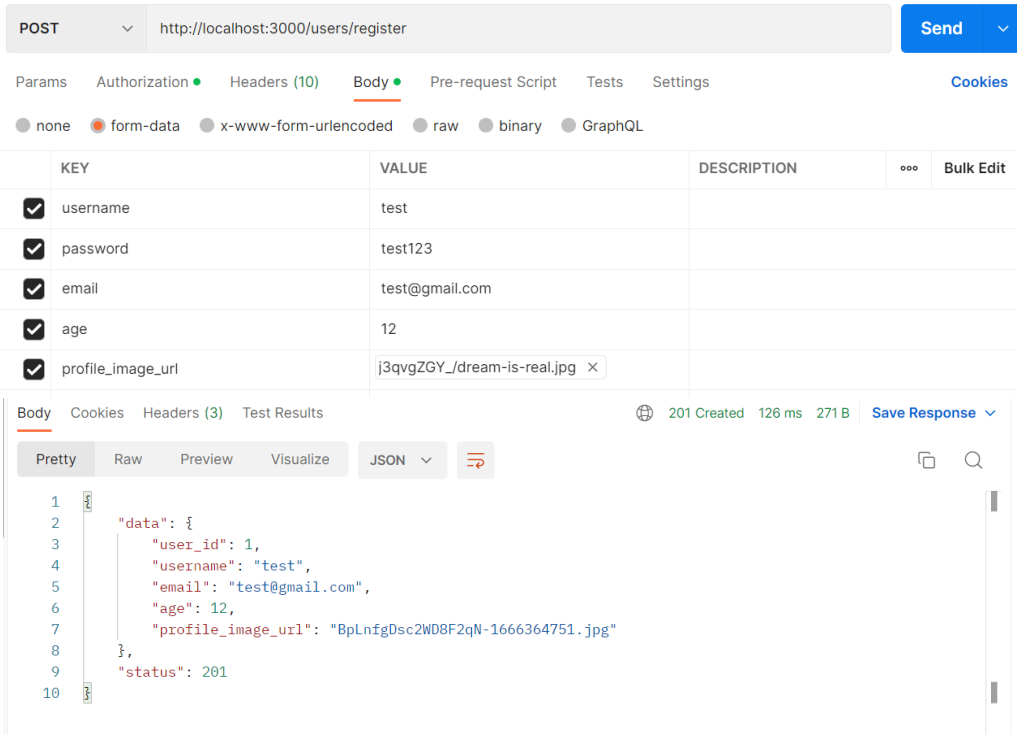
- https://www.getpostman.com/collections/5712177cc898dafe2925

Test Postman Agent

UserController

Register User

POST : http://localhost:3000/users/register



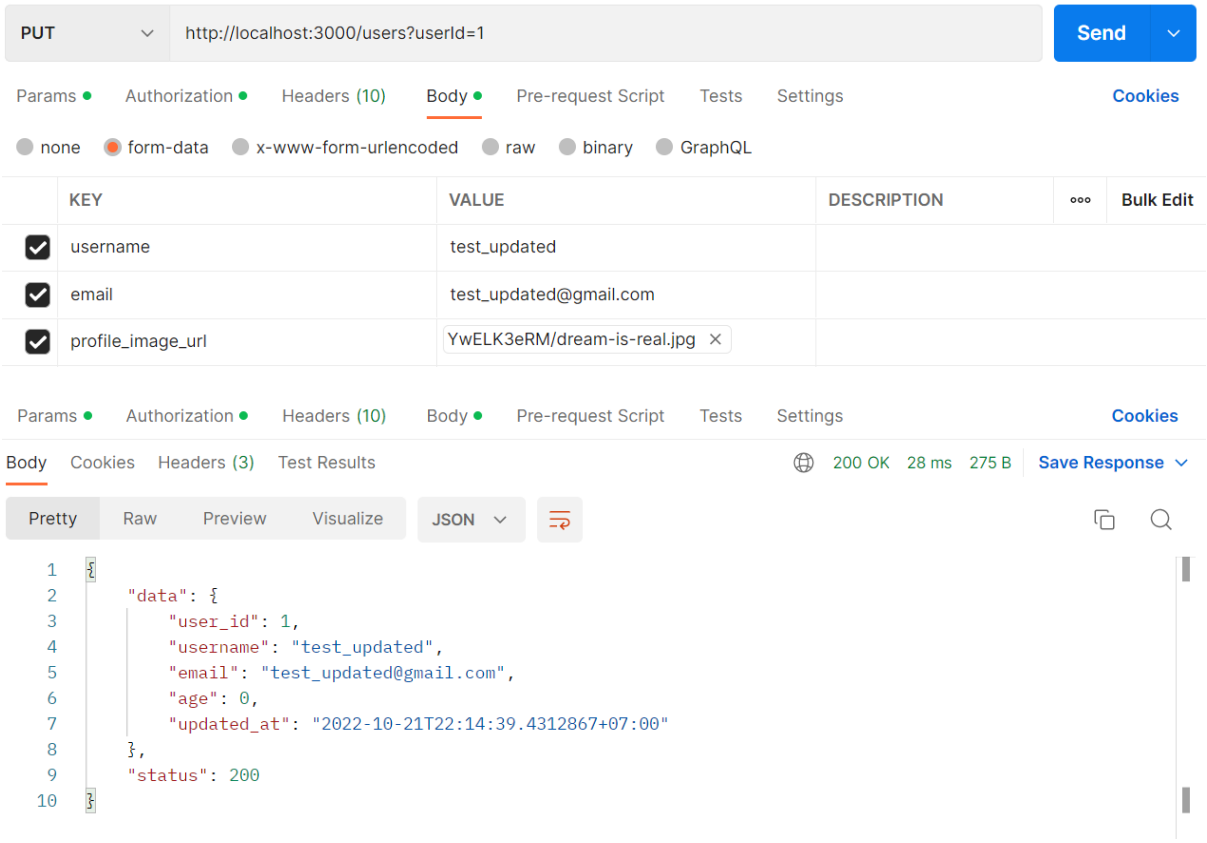
LOGIN User

POST : http://localhost:3000/users/login

Update User

PUT : http://localhost:3000/users?userId=1

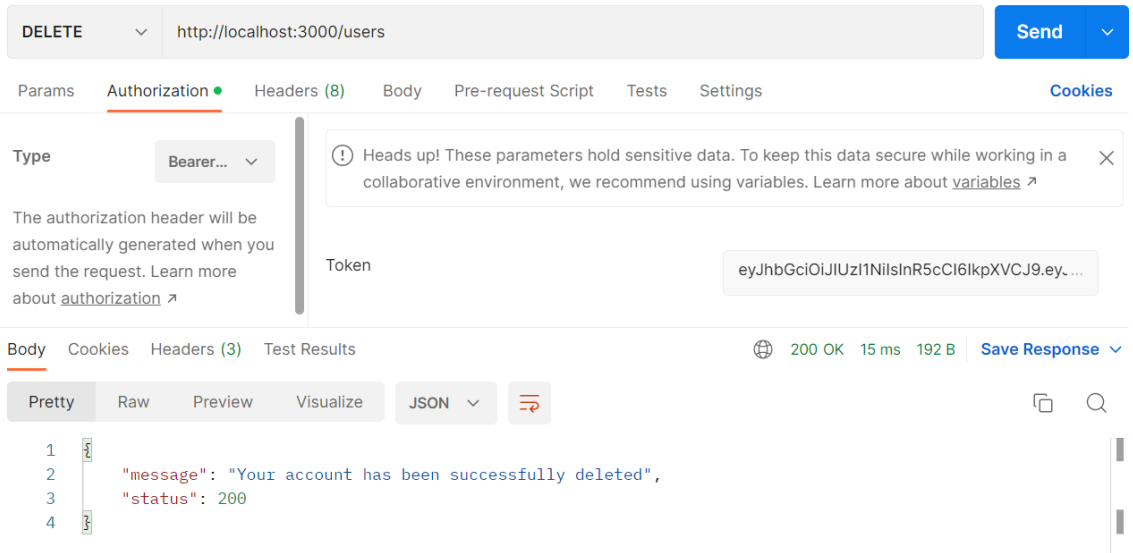
Use Authorization : Bearer Token



Delete User

DELETE : http://localhost:3000/users

Use Authorization : Bearer Token

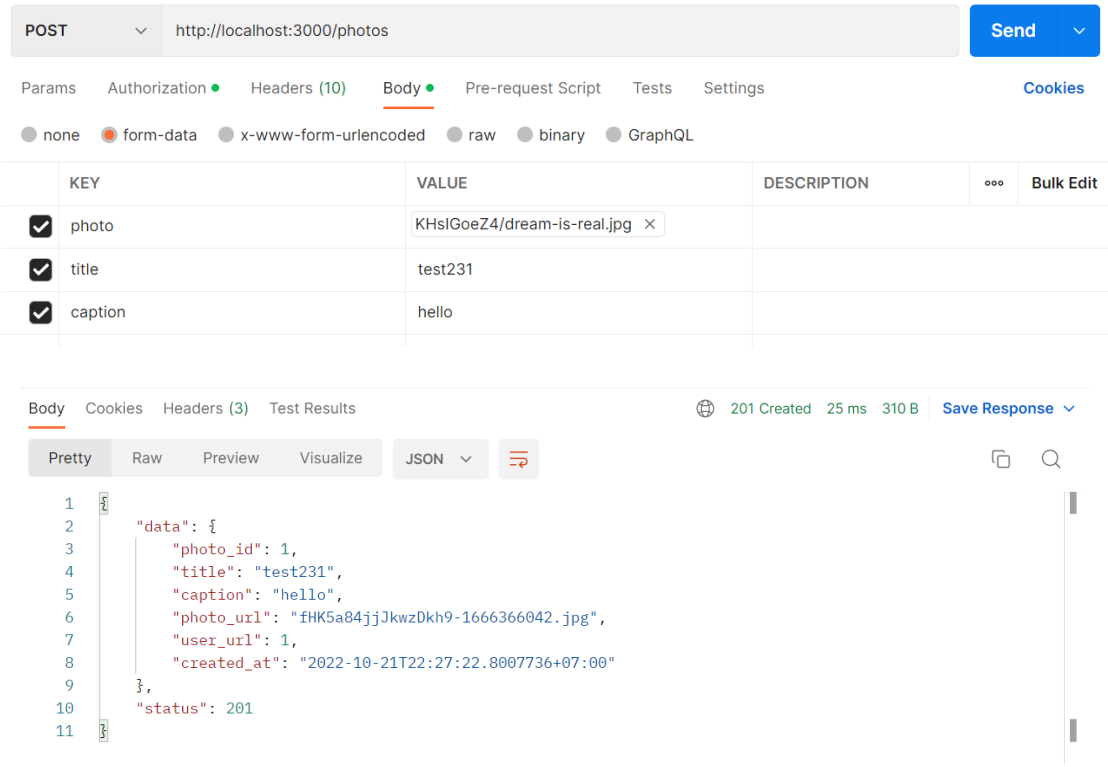


PhotoController

Post Photo

POST : http://localhost:3000/photos

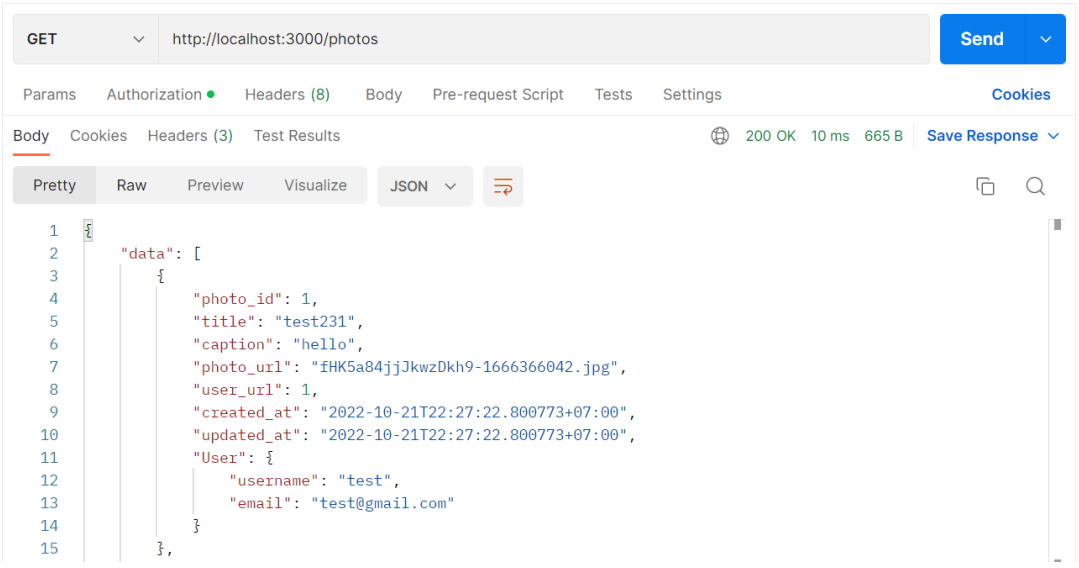
Use Authorization : Bearer Token



Get All Photo

GET : http://localhost:3000/photos

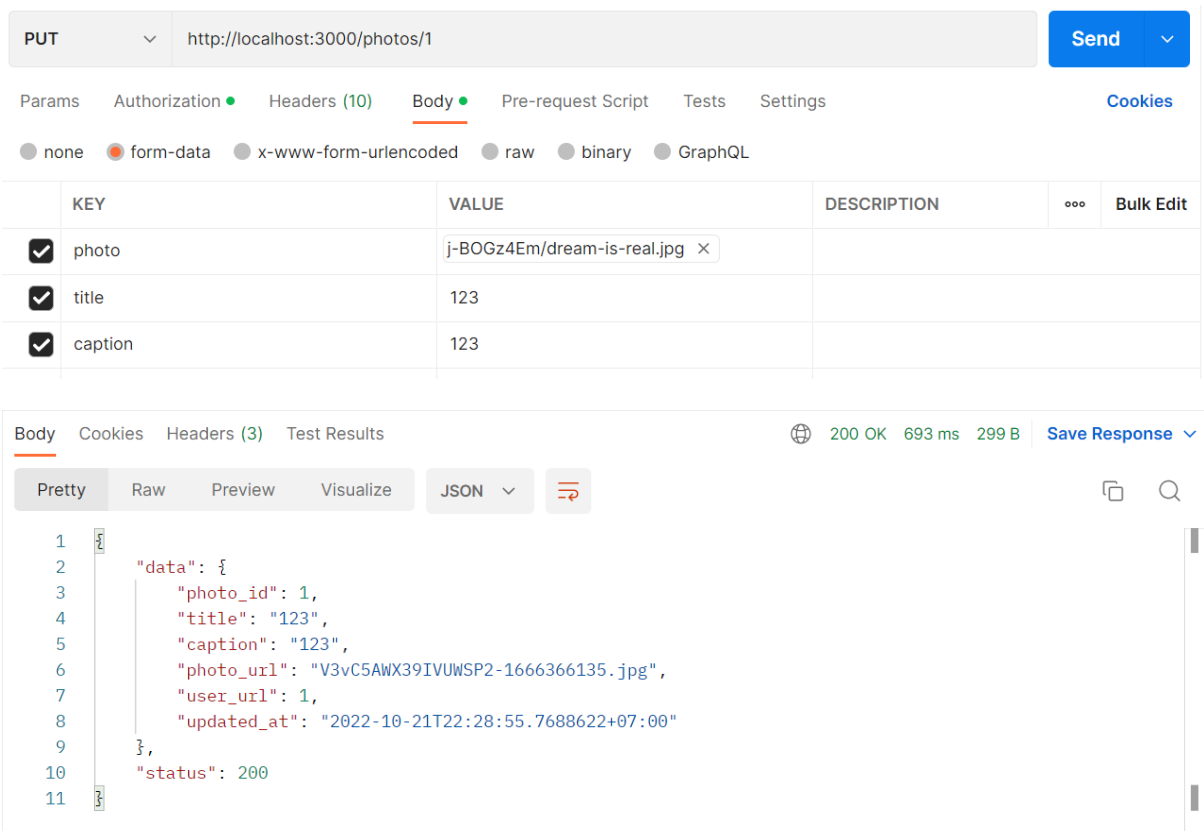
Use Authorization : Bearer Token



Update Photo

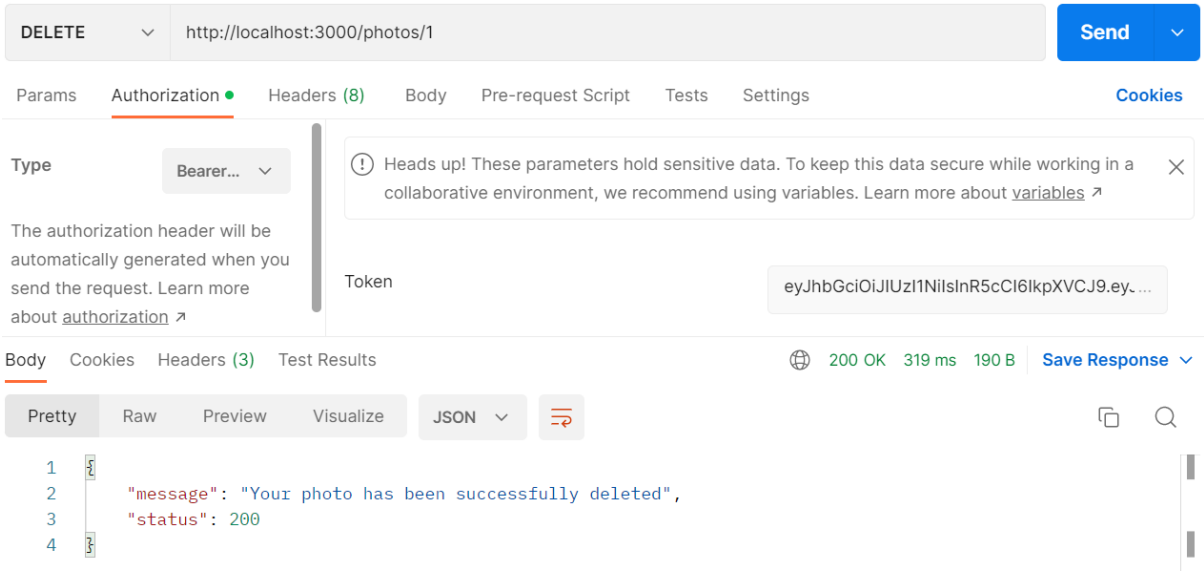
PUT : http://localhost:3000/photos/1

Use Authorization : Bearer Token



Delete Photo

Delete : http://localhost:3000/photos/1

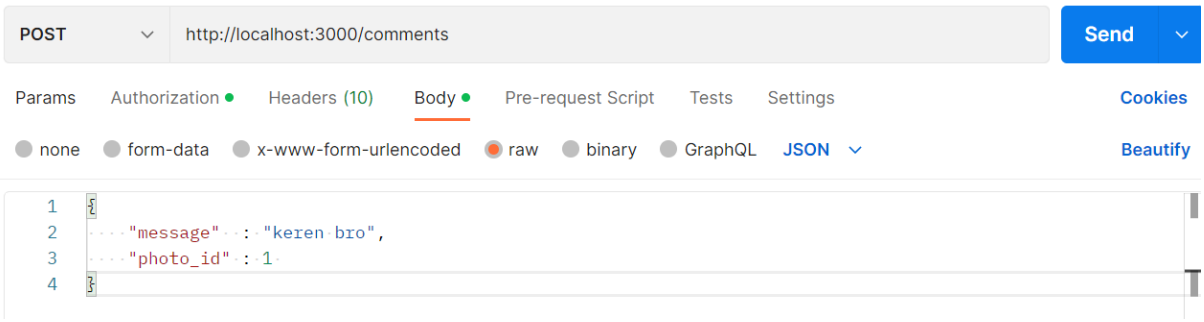


CommentController

Post Comment

POST : http://localhost:3000/comments

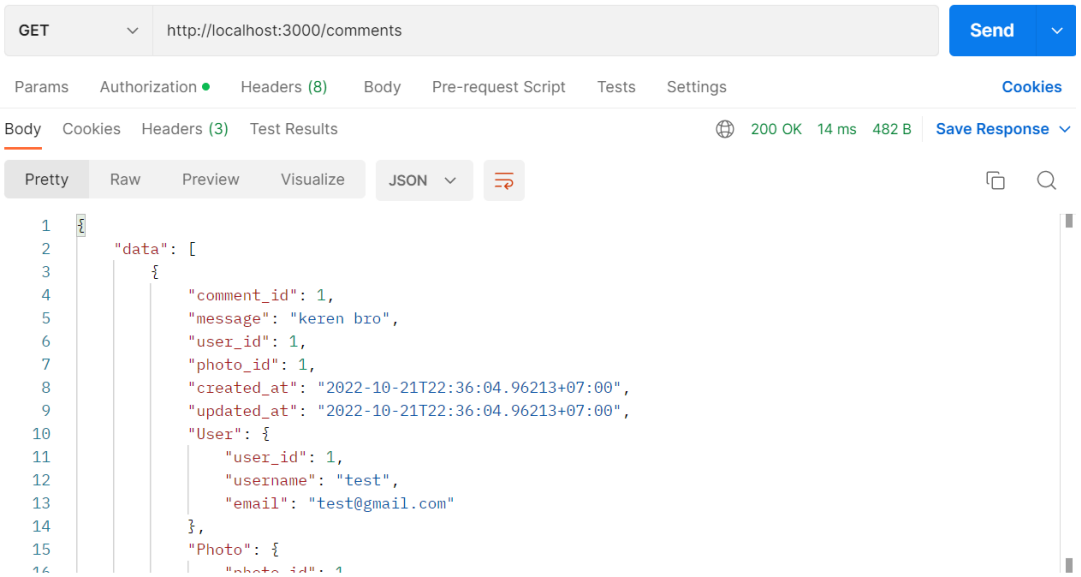
Use Authorization : Bearer Token



Get All Comments

GET : http://localhost:3000/ comments

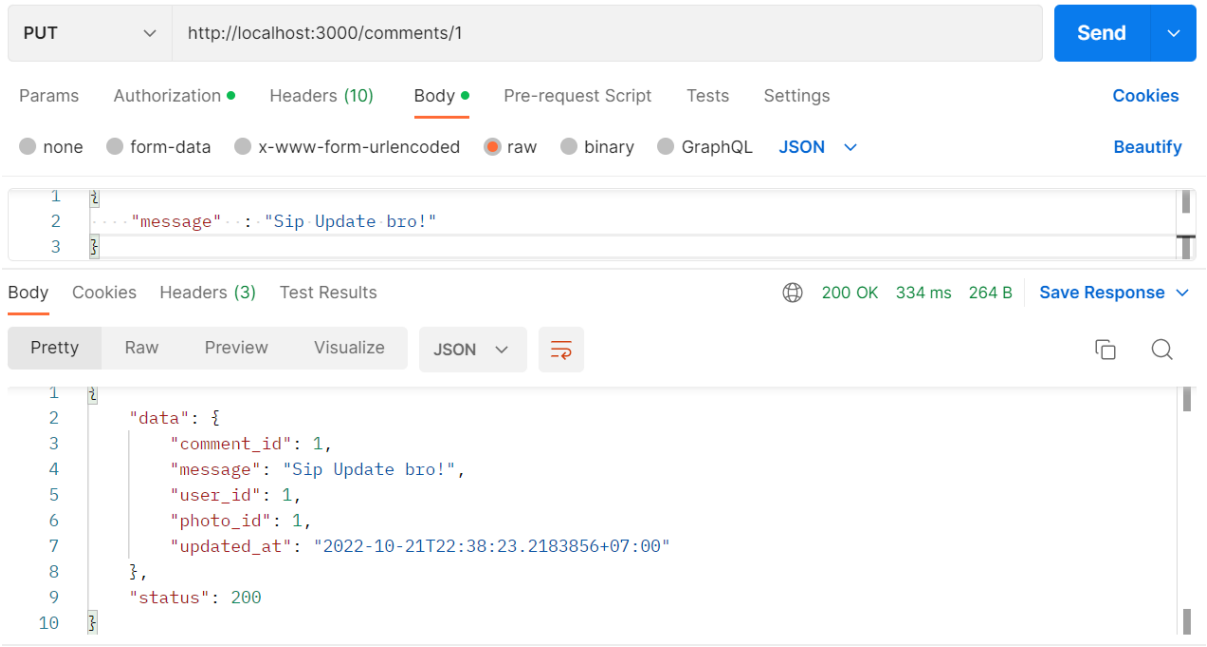
Use Authorization : Bearer Token



Update Comment

PUT : http://localhost:3000/comments /1

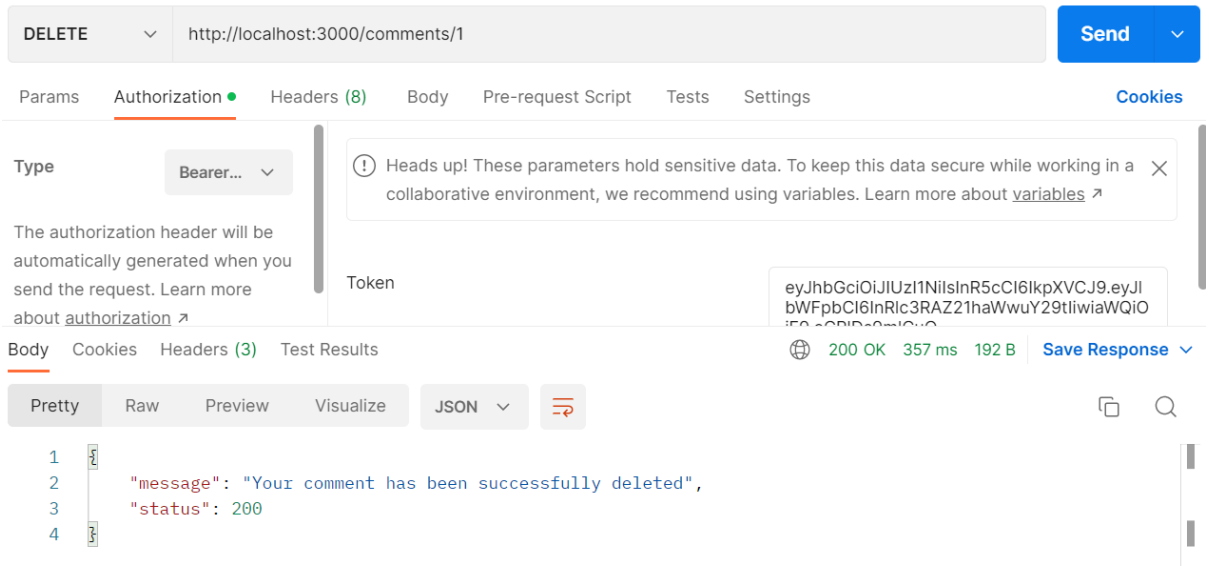
Use Authorization : Bearer Token



Delete Comment

Delete : http://localhost:3000/comments /1

Use Authorization : Bearer Token

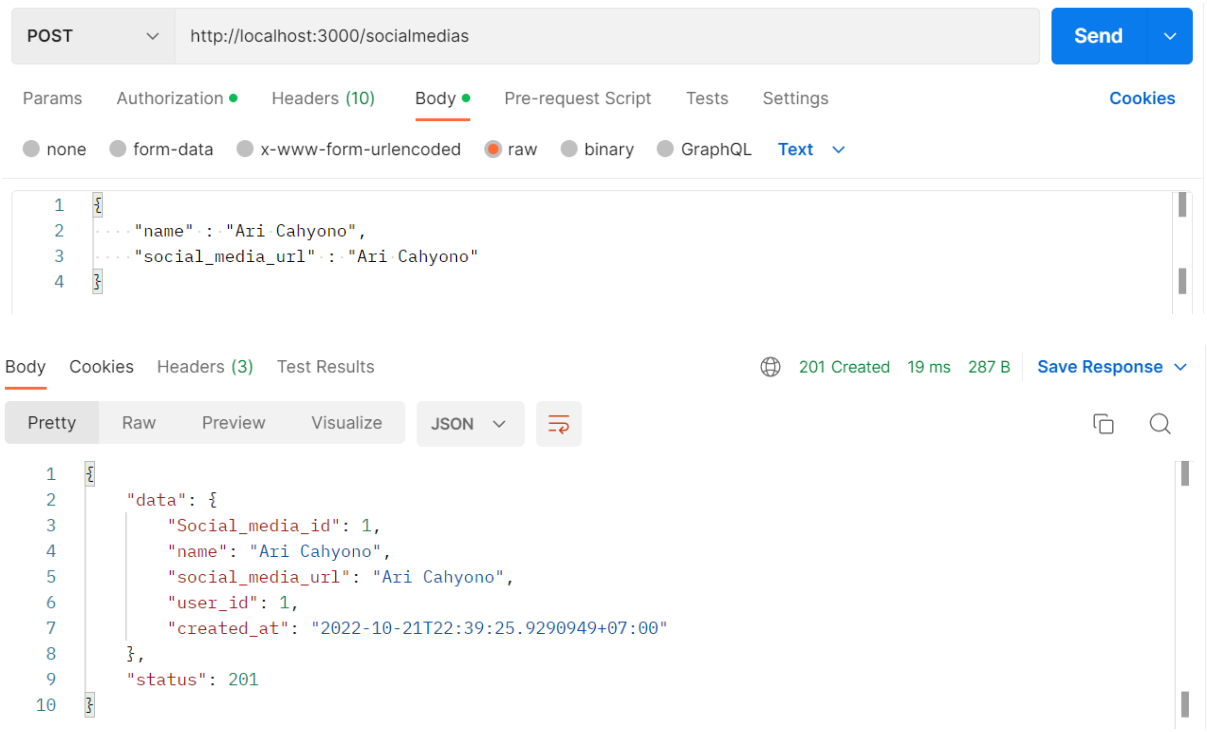


SocialmediaController

Post Social Media

POST : http://localhost:3000/socialmedias

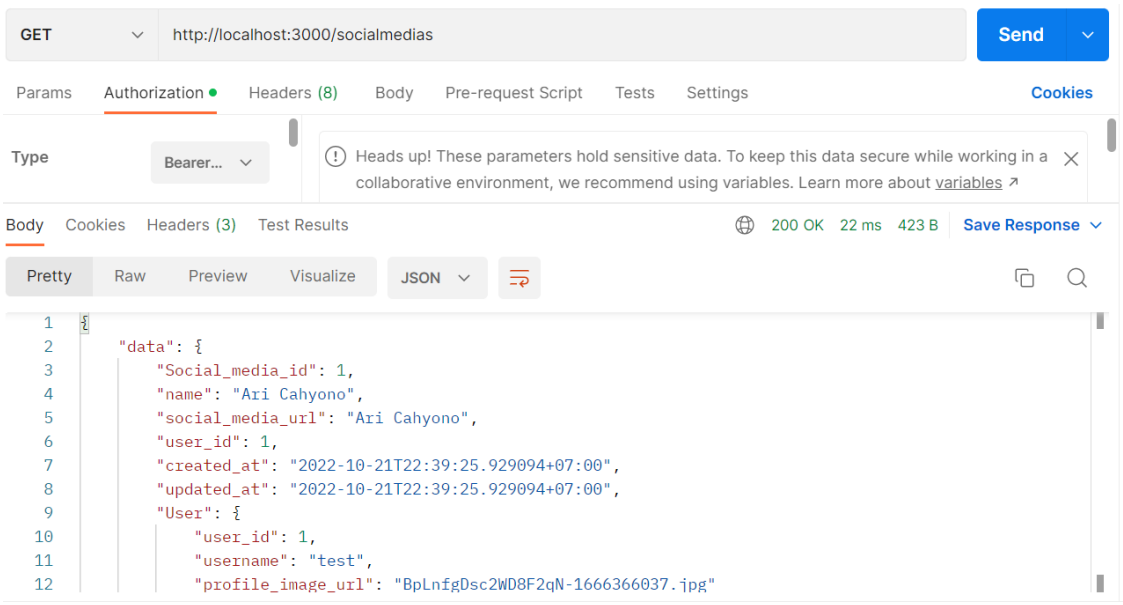
Use Authorization : Bearer Token



Get Social Media

GET : http://localhost:3000/socialmedias

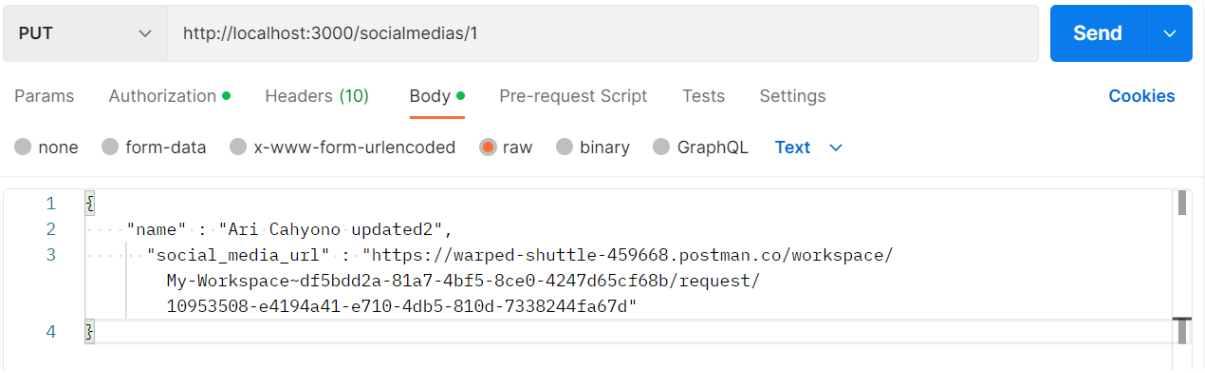
Use Authorization : Bearer Token

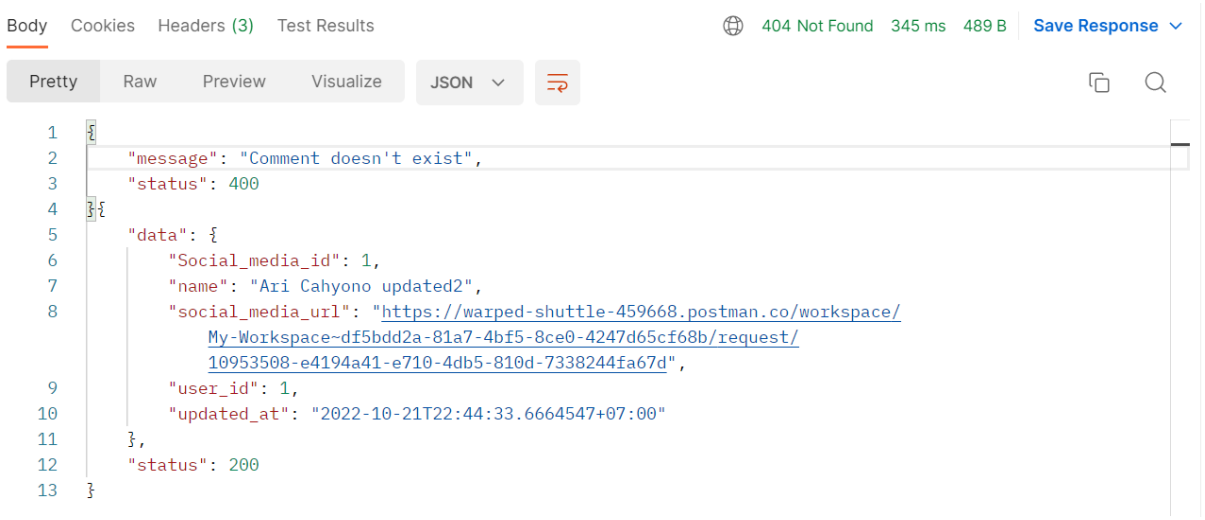


Update Social Media

PUT : http://localhost:3000/socialmedias/1

Use Authorization : Bearer Token





Delete Social Media

Delete : http://localhost:3000/socialmedias/1

Use Authorization : Bearer Token